

In [129]:

```
import numpy as np
import pandas as pd
df = pd.read_csv('C:\\Users\\niran\\Downloads\\PBI3.csv')
print df.head()
```

```
Master Serial No.      NAME OF STUDENT Roll No. Company select 1 \
0      788  Rutuja Shrinivas Abdagire  T31501  QUANTIPHI
1      789   Ankita Santosh Adchule  T31502    Fiserv
2      790   Sakshi Vinod Bajoriya  T31503  CAPGEMINI
3      791   Saurabh Digambar Bathe  T31504  CAPGEMINI
4      792   Vishakha Arjun Bhujbal  T31505    Yardi

Company select 2 Company select 3  SSCScore SSCBoard  SSCPassing  HSCScore \
0      NaN      NaN      85.00    SSC      2013    78.30
1      NaN      NaN      79.82    SSC      2013    74.62
2      Wipro      NaN      89.27    SSC      2013    75.54
3      NaN      NaN      80.18    SSC      2013    74.62
4      NaN      NaN      88.91    SSC      2013    72.46

... TE SEM 1 Percentage  TE SEM 2 Percentage  BE SEM 1 Percentage \
0 ...      66.62      66.62      68.376
1 ...      67.32      63.45      74.008
2 ...      64.68      71.10      82.016
3 ...      74.98      67.41      74.360
4 ...      60.02      67.76      75.592

Aggregate (CGPA)  Aggregate (Percentage)  LiveBacklog  DeadBacklog  YD \
0      7.73      68.05      0      0      0
1      7.54      66.31      0      5      0
2      7.54      66.31      0      4      0
3      8.19      72.10      0      1      0
4      7.74      68.12      0      4      0

Gender  Placed
0  FEMALE      1
1  FEMALE      1
2  FEMALE      1
3   MALE      1
4  FEMALE      1
```

[5 rows x 33 columns]

In [130]:

```
df.describe().T
```

Out[130]:

	count	mean	std	min	25%	50%	75%	max
Master Serial No.	72.0	824.138889	21.502247	788.000	805.7500	824.500	842.2500	862.000
SSCScore	72.0	79.589861	16.667460	8.400	77.3675	84.280	87.2000	93.820
SSCPassing	72.0	2012.833333	0.375293	2012.000	2013.0000	2013.000	2013.0000	2013.000
HSCScore	72.0	72.752083	7.331878	56.000	66.5025	74.075	78.3900	85.000
HSCPassing	72.0	2015.000000	0.000000	2015.000	2015.0000	2015.000	2015.0000	2015.000
FE SEM 1 SGPA	60.0	7.827500	1.062438	5.750	6.9800	7.880	8.6800	9.640
FE SEM 2 SGPA	60.0	7.459500	1.119590	5.450	6.6125	7.680	8.2500	9.600
SE SEM 1 SGPA	71.0	7.066338	1.056496	4.710	6.3600	7.080	7.7800	9.360
SE SEM 2 SGPA	72.0	7.051944	1.339858	2.960	6.4200	7.120	8.0050	9.400
TE SEM 1 SGPA	72.0	7.247917	1.160371	3.600	6.8000	7.410	8.0400	9.130
TE SEM 2 SGPA	72.0	7.138472	1.369569	2.210	6.5675	7.370	8.0800	8.730
BE SEM 1 SGPA	72.0	8.100972	0.948655	4.040	7.7700	8.360	8.6125	9.320
FE SEM 1 Percentage	60.0	68.880333	9.351842	50.570	61.4250	69.345	76.3800	84.830

<b>FE SEM 2 Percentage</b>	<b>count</b>	65.645667	9.849731	48.000	58.1250	67.500	72.555	84.444
<b>SE SEM 1 Percentage</b>	71.0	62.183239	9.296854	41.450	55.9850	62.300	68.4650	82.370
<b>SE SEM 2 Percentage</b>	72.0	62.057917	11.790868	26.050	56.4950	62.660	70.4450	82.720
<b>TE SEM 1 Percentage</b>	72.0	63.780694	10.212042	31.680	59.8425	65.205	70.7500	80.340
<b>TE SEM 2 Percentage</b>	72.0	62.847917	12.050049	19.450	57.7975	64.855	71.1000	76.820
<b>BE SEM 1 Percentage</b>	72.0	71.288556	8.348167	35.552	68.3760	73.568	75.7900	82.016
<b>Aggregate(CGPA)</b>	72.0	7.382500	1.021318	4.180	6.8075	7.530	8.1125	9.110
<b>Aggregate(Percentage)</b>	72.0	64.961250	8.991905	36.750	59.9100	66.260	71.3950	80.190
<b>LiveBacklog</b>	72.0	0.500000	1.727980	0.000	0.0000	0.000	0.0000	10.000
<b>DeadBacklog</b>	72.0	2.652778	2.988993	0.000	0.0000	2.000	4.0000	11.000
<b>YD</b>	72.0	0.138889	0.348257	0.000	0.0000	0.000	0.0000	1.000
<b>Placed</b>	72.0	0.777778	0.418657	0.000	1.0000	1.000	1.0000	1.000

In [131]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72 entries, 0 to 71
Data columns (total 33 columns):
Master Serial No.      72 non-null int64
NAME OF STUDENT        72 non-null object
Roll No.               72 non-null object
Company select 1       56 non-null object
Company select 2       13 non-null object
Company select 3        4 non-null object
SSCScore               72 non-null float64
SSCBoard               72 non-null object
SSCPassing             72 non-null int64
HSCScore               72 non-null float64
HSCBoard               72 non-null object
HSCPassing             72 non-null int64
FE SEM 1 SGPA          60 non-null float64
FE SEM 2 SGPA          60 non-null float64
SE SEM 1 SGPA          71 non-null float64
SE SEM 2 SGPA          72 non-null float64
TE SEM 1 SGPA          72 non-null float64
TE SEM 2 SGPA          72 non-null float64
BE SEM 1 SGPA          72 non-null float64
FE SEM 1 Percentage    60 non-null float64
FE SEM 2 Percentage    60 non-null float64
SE SEM 1 Percentage    71 non-null float64
SE SEM 2 Percentage    72 non-null float64
TE SEM 1 Percentage    72 non-null float64
TE SEM 2 Percentage    72 non-null float64
BE SEM 1 Percentage    72 non-null float64
Aggregate(CGPA)        72 non-null float64
Aggregate(Percentage)  72 non-null float64
LiveBacklog            72 non-null int64
DeadBacklog            72 non-null int64
YD                     72 non-null int64
Gender                 72 non-null object
Placed                 72 non-null int64
dtypes: float64(18), int64(7), object(8)
memory usage: 18.6+ KB
```

In [132]:

```
df.describe(include=['object'])
```

Out [132]:

	NAME OF STUDENT	Roll No.	Company select 1	Company select 2	Company select 3	SSCBoard	HSCBoard	Gender
<b>count</b>	72	72	56	13	4	72	72	72
<b>unique</b>	72	72	22	7	3	3	3	2

top	NAME OF STUDENT	Roll No.	Company select 1	Company select 2	Company select 3	SSCBoard	HSCBoard	Gender
	Shinde	T3162	L&T INFOTECH	CAI GEMINI	ITAA	SSC	HSC	FEMALE
freq	1	1	10	6	2	64	57	47

In [133]:

```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

placedcounts=df['Placed'].value_counts()
print("Placed count is:",placedcounts[1])
total = placedcounts[0]+placedcounts[1]
print("Total count is:",total)
# Using matplotlib pie chart and label the pie chart

plt.pie(placedcounts,labels=['placed','not placed']);
```

('Placed count is:', 56)  
('Total count is:', 72)



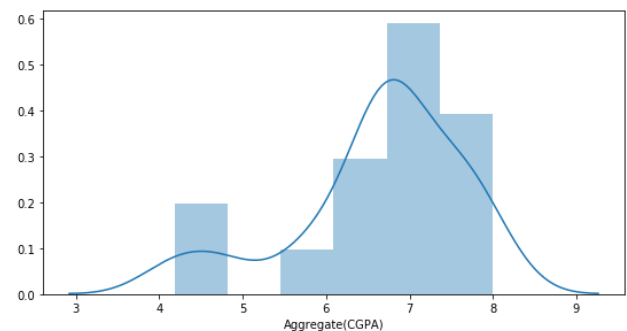
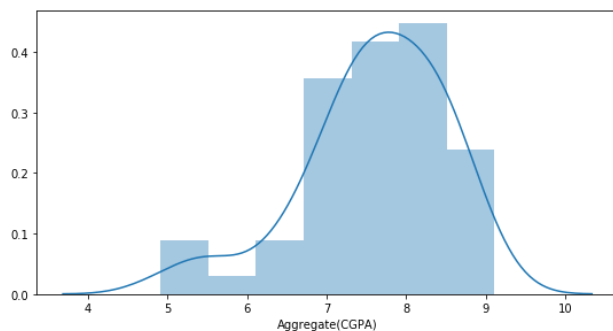
In [134]:

```
placedddf = df[df['Placed']==1]
notplacedddf =df[df['Placed']==0]
```

In [135]:

```
fig = plt.figure(figsize=(20,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

# Tell pointplot to plot on ax1 with the ax argument (satisfaction level)
sns.distplot(placedddf['Aggregate (CGPA)'],ax = ax1);
# Tell the factorplot to plot on ax2 with the ax argument (satisfaction level)
sns.distplot(notplacedddf['Aggregate (CGPA)'],ax = ax2);
```

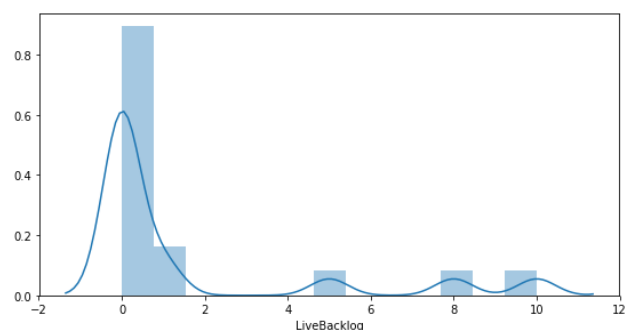
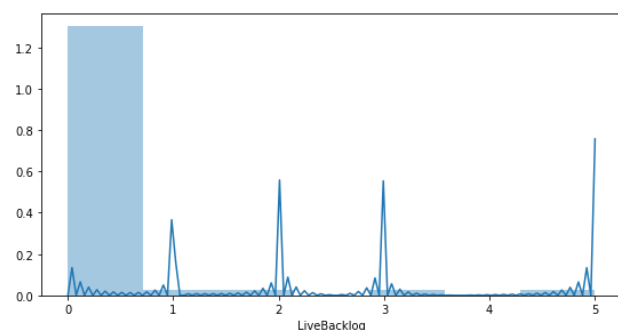


In [136]:

```
fig = plt.figure(figsize=(20,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)
```

```
# Tell pointplot to plot on ax1 with the ax argument
sns.distplot(placeddf['LiveBacklog'], kde=True, ax=ax1);

# Tell the factorplot to plot on ax2 with the ax argument
sns.distplot(notplaceddf['LiveBacklog'], kde=True, ax=ax2);
```

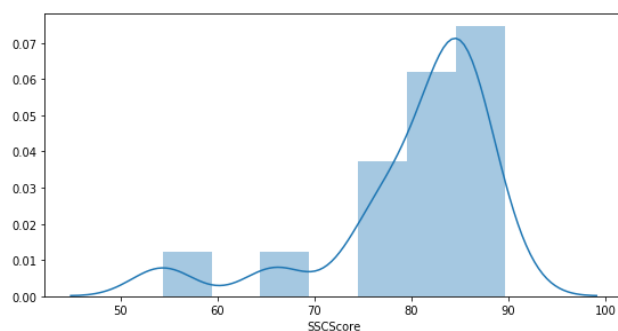
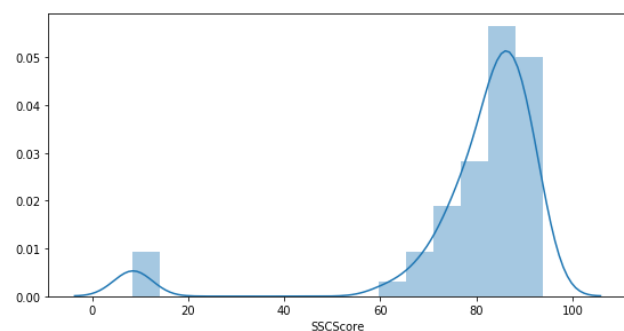


In [137]:

```
# Create a figure instance, and the two subplots
fig = plt.figure(figsize=(20,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

sns.distplot(placeddf['SSCScore'], kde=True, ax=ax1);

sns.distplot(notplaceddf['SSCScore'], kde=True, ax=ax2);
```

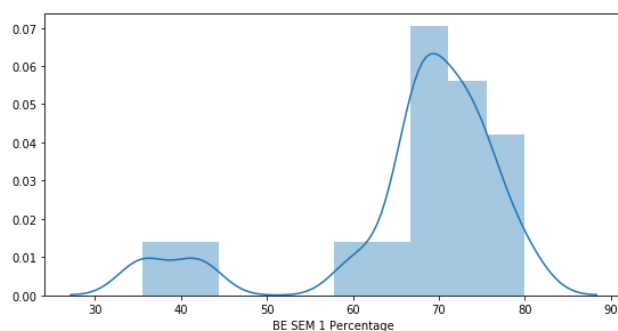
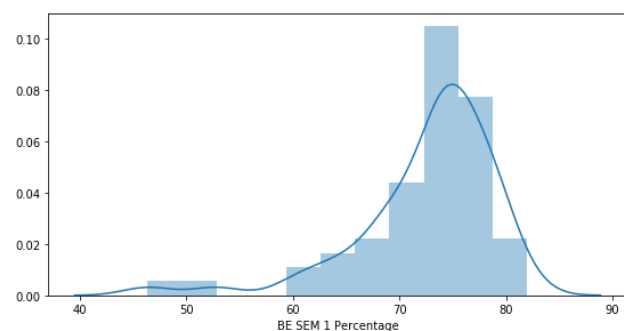


In [138]:

```
# Create a figure instance, and the two subplots
fig = plt.figure(figsize=(20,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

sns.distplot(placeddf['BE SEM 1 Percentage'], kde=True, ax=ax1);

sns.distplot(notplaceddf['BE SEM 1 Percentage'], kde=True, ax=ax2);
```



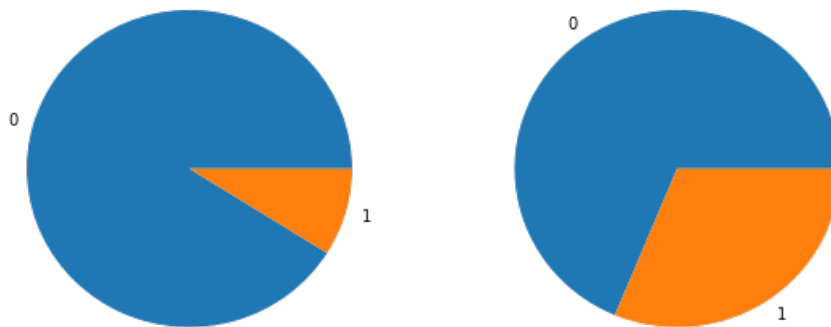
In [139]:

```
#create a figure with two subplots
```

```
fig = plt.figure(figsize=(10,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

# Do the value counts of work accident
placedYDcounts = placedddf['YD'].value_counts()
notplacedYDcounts = notplacedddf['YD'].value_counts()

# plot each pie chart in a separate subplot
ax1.pie(placedYDcounts,labels=placedYDcounts.index);
ax2.pie(notplacedYDcounts,labels=notplacedYDcounts.index);
```

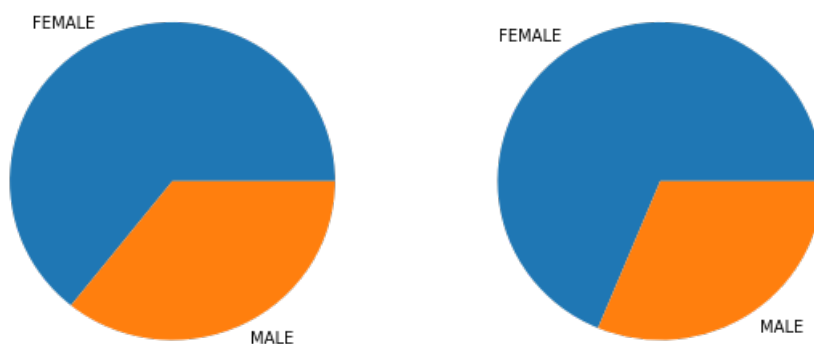


In [140]:

```
# create a figure with two subplots
fig = plt.figure(figsize=(10,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

placedGendercounts = placedddf['Gender'].value_counts()
notplacedGendercounts = notplacedddf['Gender'].value_counts()

# plot each pie chart in a separate subplot
ax1.pie(placedGendercounts,labels=placedGendercounts.index);
ax2.pie(notplacedGendercounts,labels=notplacedGendercounts.index);
```



In [141]:

```
df.drop(['Company select 2','Company select 3'],inplace = True, axis = 1)
```

In [142]:

```
df.drop(['Master Serial No.','NAME OF STUDENT','Roll No.'],inplace = True, axis = 1)
```

In [143]:

```
df.drop(['Company select 2','Company select 3'],inplace = True, axis = 1)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72 entries, 0 to 71
Data columns (total 28 columns):
Company select 1      56 non-null object
SSCScore              72 non-null float64
SSCBoard              72 non-null object
SSCPassing            72 non-null int64
HSCScore              72 non-null float64
HSCBoard              72 non-null object
HSCPassing            72 non-null int64
FE SEM 1 SGPA         60 non-null float64
FE SEM 2 SGPA         60 non-null float64
SE SEM 1 SGPA         71 non-null float64
SE SEM 2 SGPA         72 non-null float64
TE SEM 1 SGPA         72 non-null float64
TE SEM 2 SGPA         72 non-null float64
BE SEM 1 SGPA         72 non-null float64
FE SEM 1 Percentage   60 non-null float64
FE SEM 2 Percentage   60 non-null float64
SE SEM 1 Percentage   71 non-null float64
SE SEM 2 Percentage   72 non-null float64
TE SEM 1 Percentage   72 non-null float64
TE SEM 2 Percentage   72 non-null float64
BE SEM 1 Percentage   72 non-null float64
Aggregate (CGPA)      72 non-null float64
Aggregate (Percentage) 72 non-null float64
LiveBacklog           72 non-null int64
DeadBacklog           72 non-null int64
YD                    72 non-null int64
Gender                 72 non-null object
Placed                 72 non-null int64
dtypes: float64(18), int64(6), object(4)
memory usage: 15.8+ KB
```

```
In [144]:
```

```
df.SSCBoard = df.SSCBoard.map({'SSC':0, 'CBSE':1, 'ICSE':2})
df.HSCBoard = df.HSCBoard.map({'HSC':0, 'CBSE':1, 'MSBTE':2})
```

```
In [145]:
```

```
df.drop(['Company select 1'], inplace = True, axis = 1)
```

```
In [146]:
```

```
df.Gender = df.Gender.map({'MALE':0, 'FEMALE':1})
```

```
In [147]:
```

```
df = df.drop(['SSCPassing', 'HSCPassing'], axis=1)
```

```
In [148]:
```

```
df = df.fillna(df.mean())
```

```
In [149]:
```

```
# X = df[['SSCScore', 'HSCScore', 'SSCBoard', 'FE SEM 1 SGPA']]
# X = X.fillna(X.mean())
# X.isnull()
```

```
In [150]:
```

```
# X = df.drop(['Placed'], axis=1)
y = df['Placed']

print X.head()
print len(X)
```

```

SSCScore  HSCScore  SSCBoard  FE SEM 1 SGPA
0      85.00      78.30          0          8.60
1      79.82      74.62          0          8.08
2      89.27      75.54          0          7.80
3      80.18      74.62          0          8.40
4      88.91      72.46          0          8.68
72

```

In [172]:

```

import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import preprocessing, neighbors, svm
from sklearn.metrics import accuracy_score

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
print(len(y_train) + len(y_test))

```

72

In [173]:

```

from sklearn.metrics import accuracy_score, log_loss
from sklearn.ensemble import RandomForestClassifier
from sklearn import tree
randomclf = RandomForestClassifier()
randomclf.fit(X_train, y_train)

pred = randomclf.predict(X_test)
accuracy = accuracy_score(pred, y_test)
print "Accuracy Random Forest - ", accuracy

```

Accuracy Random Forest - 0.5333333333333333

In [174]:

```

clf = svm.SVC()
# Fit the svm object with train data
clf.fit(X_train, y_train)

```

Out[174]:

```

SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
    kernel='rbf', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)

```

In [175]:

```

pred = clf.predict(X_test)
accuracy = accuracy_score(pred, y_test)

print "Accuracy using SVC classifier - ", accuracy

```

Accuracy using SVC classifier - 0.9333333333333333